(19)日本国特許庁（ＪＰ）　(12)**公開特許公報**（Ａ）　(11)特許出願公開番号

特開平10－154084

(43)公開日　平成10年(1998)６月９日

(51)Int.Cl.[6]	識別記号	ＦＩ	
Ｇ０６Ｆ　11/28	３４０	Ｇ０６Ｆ　11/28	３４０Ｃ

審査請求　未請求　請求項の数３　ＯＬ　（全　８　頁）

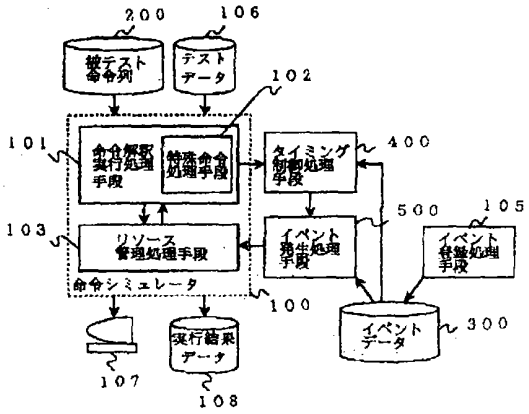
(21)出願番号　特願平8－311867

(22)出願日　平成８年(1996)11月22日

(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地
(72)発明者　芳野　泰成
神奈川県秦野市堀山下１番地　株式会社日立製作所汎用コンピュータ事業部内
(72)発明者　中田　孝広
神奈川県秦野市堀山下１番地　株式会社日立製作所汎用コンピュータ事業部内
(74)代理人　弁理士　鈴木　誠

(54)【発明の名称】　命令シミュレーション方法および命令シミュレーションシステム

(57)【要約】
【課題】　非同期イベントの発生を、任意の制御性と再現性を確保しつつ簡便に実施可能とする。
【解決手段】　命令シミュレータ１００の命令実行処理とは非同期に発生するイベントのデータ３００を登録する手段５００を設ける。また、被テスト命令列２００には、非同期イベントの発生タイミングを制御するための特殊命令を予め埋め込んでおく。命令シミュレータ１００は、命令処理が特殊命令に達するとタイミング制御処理手段４００をコールする。該処理手段４００は、特殊命令の指示に基づき、非同期イベントの発生タイミングに達した時点でイベント発生処理手段５００をコールする。該処理手段５００は、イベントデータ３００を読み込み、命令シミュレータ１００上に非同期イベントを発生させる。



1

【特許請求の範囲】

【請求項1】　被テスト命令列とテストデータを入力し、前記被テスト命令列を解釈実行してハードウェア動作を模擬する命令シミュレーション方法において、
前記被テスト命令列に特殊命令を予め埋め込み、前記特殊命令によって、命令実行処理とは非同期に発生するイベントの発生タイミングを任意に制御し、非同期イベントを前記命令実行処理内に発生させることを特徴とする命令シミュレーション方法。

【請求項2】　被テスト命令列とテストデータを入力とし、前記被テスト命令列を解釈実行してハードウェア動作を模擬する命令シミュレータと、
命令実行処理とは非同期に発生するイベントのデータを登録するイベント登録手段と、
前記被テスト命令列に予め埋め込まれた特殊命令に達した際に前記命令シミュレータからコールされて、前記特殊命令の指示に基づき、命令実行処理と非同期イベントの発生タイミングを制御するタイミング制御手段と、
前記タイミング制御処理手段から非同期イベントの発生許可を受けて、前記登録されているイベントデータを読み込み、前記命令シミュレータに非同期イベントを発生させるイベント発生処理手段と、を有することを特徴とする命令シミュレーションシステム。

【請求項3】　各々非同期に被テスト命令列を実行する複数の命令シミュレータを具備し、イベント登録手段は、非同期イベントを発生させる一方の命令シミュレータでの命令実行処理とは非同期に実行される他方の命令シミュレータのイベントデータを登録することを特徴とする命令シミュレーションシステム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は命令シミュレーション方法および命令シミュレーションシステムに係り、特に、特定のハードウェアの機能や構造を意識した命令列からなる被テスト命令列を逐次模擬する命令シミュレータを利用したソフトウェア検証方法およびシステムに関し、さらに詳しくは、非同期動作を含んだハードウェア動作に適合したソフトウェアの検証方法およびシステムに関する。

【０００２】

【従来の技術】近年のコンピュータやコントローラを代表とする情報処理装置においては、一般にその機能は、ハードウェアとその上で実行されるハードウェアを直接制御するソフトウェアによって実現されている。例えば、汎用コンピュータにおいては、複雑な機能を持った命令の処理や、周辺装置とのデータのやり取りに対して、ハードウェアだけでなくマイクロプログラムと呼ばれるハードウェアを直接制御するソフトウェアを用いて実現されている。また、民生用や産業用のコントローラにおいては、汎用的なＣＰＵをコアとして、各機能がその上で実行されるソフトウェアによって実現されている。なお、これらのソフトウェアは一般に組み込みソフトウェアと呼ばれる。さらには、最近のＲＩＳＣプロセッサ用のコンパイラのように、アプリケーションプログラムでありながら、ハードウェアを意識して開発され、その製品の性能を左右するようなハードウェアの実現方式に依存したソフトウェアも存在する。さらには、ハードウェアの検証を目的として、個々のハードウェアの実現機能の動作試験を行うハードウェアの構成方式に依存したテストプログラムも開発されている。

2

【０００３】以上のようなハードウェアを直接制御する、あるいはハードウェアの実現方法に依存したソフトウェアの開発検証においては、一般のアプリケーションソフトウェア開発にはない大きな問題がある。それは、開発したソフトウェアを実行する環境が早期に得られないという点である。ハードウェアとソフトウェアの双方を含めた全体の開発期間の短縮のために、一般にハードウェアの開発製造が終了してからソフトウェアの開発を実施するのではなく、並行して開発することが要求される。しかし、開発対象のソフトウェアはハードウェアの構成方式に依存した命令列となっているため、ハードウェアの開発が終了するまでの期間において、ソフトウェアの検証を行う環境を得ることができない。

【０００４】これに対して、ハードウェアが製造されるまでは、一般に命令シミュレータによるシミュレーションによって、これらのソフトウェアの開発・検証が行われる。ここで、命令シミュレータとは、シミュレータ内にハードウェア資源や動作のモデルを保持し、被テストプログラムを逐次解釈実行しながら、ハードウェア動作を模擬し、その実行結果として得られたハードウェア資源の数値等により、被テストプログラムが仕様通りに実現されているか否かを検証するものである。命令シミュレータには、基本的に被テストプログラムを逐次、実行解釈するモデルが採用される。また命令シミュレータ内での時間管理は命令単位であり、１命令の解釈・実行を１時間単位としたモデルが採用される。これについては、例えば、特開平０３－１０２５４０号公報に記載のように、プログラム内の指定された命令実行状態で命令実行を停止し正常状態になっていることを逐次確認するプログラムのテスト方式等が知られている。

【０００５】

【発明が解決しようとする課題】一般にハードウェアは、命令処理を逐次実行する部分と、非同期に動作する周辺論理の部分とに大別することができる。また、マルチプロセッシング構成等においては、複数の逐次実行する部分が互いに非同期に実行されている。この時、複数の逐次処理間にお互いにデータのやり取りがあった場合、一つの逐次処理部に着目すれば、他からのデータの書き込みは非同期に発生するイベントとして扱うことができる。そこで、被テストプログラムが先に述べたハー

ドウェアの実現方式に依存し、ハードウェアを直接制御するような場合において、逐次処理を行っている命令シミュレータ上での非同期イベントの発生方法が問題になる。

【０００６】通常の命令シミュレータにおいて、単純に非同期イベントを発生させることは容易である。しかし、命令シミュレータがソフトウェアの検証を目的としているため、その非同期イベントの発生タイミングは操作者の制御下におかれていなければならない。即ち、シミュレーションをする場合、テストケースに基づいてテストを実施するが、非同期イベントに対しても当然テストケースに従って、意図したタイミングで発生できなければならない。また、デバッグ等を行って、同じ条件でシミュレーションを再実行させた場合には、前回と同じタイミングで発生しなければならない。つまり、再現性が要求される。以上より命令シミュレータにおいては、単に非同期イベントを発生させればよいだけでなく、そのイベントの発生のタイミングを意図したものにする制御性と、非同期イベントの発生が同一条件で同じに発生する再現性を確保することが要求される。

【０００７】これに対しては、次の方法が考えられる。例えば、非同期イベントを発生させたいタイミングに該当する被テストプログラム中の命令にブレークポイントを設定し、そこまで命令処理を実行させシミュレーションを中断させた後、非同期イベントを発生させる処理を手操作によって実行し、あらためてシミュレーションを再開させる方法である。これにより、意図したタイミングでの非同期イベントを発生可能で、かつ再現性も確保できる。しかしながら、ブレークポイントの設定やシミュレーションの停止、再実行、非同期イベントの発生指示等の操作が必要になる。ソフトウェアの検証においては、同様のケースを何度も実行しなければならない場合が多く、上記の方法ではその都度上記の操作が必要となり、実用には供しない。

【０００８】本発明の目的は、非同期イベントの発生を、任意の制御性と再現性を確保しながら簡便に実施可能とする命令シミュレーション方法およびシステムを提供することにある。

【０００９】

【課題を解決するための手段】本発明では、まず、発生させたい非同期イベントの情報を登録する手段を設け、該手段を命令シミュレーションとは非同期に実行し、非同期イベントの情報を格納・登録する。また、被テスト命令列には、非同期イベントの発生タイミングを制御することを目的とした特殊命令を予め埋め込んでおく。命令シミュレータにより被テスト命令列の命令実行処理が進み、前記特殊命令に達した際に、非同期イベントの発生タイミングを制御する処理手段をコールする。このタイミング制御処理手段は、前記特殊命令の命令オペランドを読み込み、発生させる非同期イベントの種類と発生

タイミングを取得する。そして、命令シミュレーションが発生タイミングに達した時点でイベント発生処理手段をコールする。このイベント発生処理手段は先に格納してある非同期イベントの情報を読み出し、命令シミュレータ上に非同期イベント発生させる。この時、発生させるべき非同期イベントの情報が未だ格納されていない場合は、シミュレーションを停止し、登録手段により該当する非同期イベントが登録されるのを待ち、そのイベントを発生させ、シミュレーションを再スタートさせる。一方、命令実行が発生タイミングに達した際に、すでに該当する非同期イベントが登録されている場合は、ただちに非同期イベントを命令シミュレータ上に発生させる。

【００１０】以上のようにして、操作者は特殊命令のオペランドにより任意のタイミングで非同期イベントを発生させることが可能となる。併せて、これらの条件が同一の場合は、一意にタイミングを決定することができる。さらに予め被テストプログラムに特殊命令を埋めこみオペランドデータを変えるだけの容易な操作で非同期イベントの発生を制御することが可能となる。これにより、非同期イベントを操作者の意図した任意のタイミングで発生可能とする制御性を持つだけでなく、テスト環境に必要な再現性を確保することが可能となり、簡便な操作で命令シミュレーションを可能とする。

【００１１】

【発明の実施の形態】図１に、本発明の第１の実施例である命令シミュレーションシステムの構成図を示す。本命令シミュレーションシステムは、被テスト命令列２００とテストデータ１０６を入力とし、該被テスト命令列２００をテストデータ１０６の条件下でシミュレーションして結果を表示装置１０７に表示あるいは実行結果データ１０８として出力する命令シミュレータ１００と、非同期イベントの情報をイベントデータ３００に登録するイベント登録処理手段１０５、非同期イベントの発生タイミングを制御・調整するタイミング制御処理手段４００、及び命令シミュレータ１００上に非同期イベントを発生させるイベント発生処理手段５００より構成される。非テスト命令列２００には、予め非同期イベントの発生タイミングをコントロールすることを目的とした特殊命令が挿入されている。

【００１２】図１において、命令シミュレータ１００は、特殊命令処理手段１０２を除くと、従来の通常の命令シミュレータと同じであり、命令を解釈しシミュレーションする命令解釈実行処理手段１０１と、ハードウェアの各リソースの値の保持、並びに読み書きを管理するリソース管理処理手段１０３、その他、デバッグのためのユーザインターフェース処理手段や結果を出力する処理手段等から構成される。ここで、命令解釈実行処理手段１０１に特殊命令処理手段１０２が組み込まれる。該命令シミュレータ１００の特殊命令処理手段１０２を除

いた構成は従来の通常の命令シミュレータと同じであるので、図１においては詳細構成は省略している。　本命令シミュレーションシステムでは、被テスト命令列２００をシミュレーションする命令シミュレータ１００とは非同期に実行されるイベント登録処理手段１０５にて、非同期イベントの情報をイベントデータ３００に登録する。ここで、イベント登録処理手段１０５は、非同期イベントの情報を登録できるものであれば何でもよく、例えば、オペレータ自身が登録する場合は、キーボードなどがイベント登録処理手段の役割をはたす。また、後述の図６に示すように、非同期に実行されている他の命令シミュレータに起動されて非同期イベントの情報を自動的に生成・登録するソフトウェア／ハードウェアでもよい。命令シミュレータ１００が被テスト命令列２００を逐次シミュレーションして、特殊命令に達した時に、命令解釈実行処理手段１０１に組み込まれた特殊命令処理手段１０２が該特殊命令を実行し、タイミング制御処理手段４００を起動して、該命令オペランドで指示された非同期イベントのタイプ（種類）と発生タイミングのカウント値を渡す。タイミング制御処理手段４００は、その発生タイミングになるまで時間を調整し、発生タイミングになった時に、イベント発生処理手段５００を起動する。イベント発生処理手段５００は、該当する非同期イベントの情報をイベントデータ３００より読み出し、命令シミュレータ１００のリソース管理処理手段１０３に働き掛けて、非同期イベントを命令シミュレータ１００上に発生させる。

【００１３】なお、タイミング制御処理手段４００では、非同期イベントの発生タイミングに達した時に、外部から該当非同期イベントの情報がイベントデータ３００にない場合は、命令シミュレーションを停止させ、該当する非同期イベントの情報がイベントデータ３００に登録されるまで待ち続け、登録された時点でイベント発生処理手段５００を起動する。

【００１４】このように、本命令シミュレーションシステムでは、ユーザが被テスト命令列に埋め込んだ特殊命令の命令オペランドを変えることにより、任意のタイミングで非同期イベントを発生させることが可能となる。また、タイミング制御処理手段４００が、前記特殊命令に従って発生タイミングを一意に決定し、命令シミュレータ１００の実行と非同期イベントの発生を管理するため、再現性も保証される。さらには、特殊命令の命令オペランドをセットするだけの簡便な操作で、上記の非同期イベントの発生タイミングをコントロールすることが可能となる。

【００１５】以下、図２乃至図５を併用して、図１に示す命令シミュレーションシステムの各部について更に詳しく説明する。

【００１６】図２に、本命令シミュレーションシステムの入力となる被テスト命令列２００及びそれに埋め込まれた非同期イベントの発生タイミングをコントロールする特殊命令２０１の一例を示す。ここで、被テスト命令列２００は、命令シミュレータ１００によって逐次解釈実行される命令の流れを模式的に表したものであり、実際の主記憶上のプログラムを示したものではない。また、本例の場合は、命令長は３２ビットの固定長としている。

【００１７】図２において、被テスト命令列（被テストプログラム）２００には、非同期イベントの発生タイミングを制御するための特殊命令２０１が予め挿入されている。特殊命令２０１は、オペコード２０２と、命令オペランドとしてイベント種類２０３とカウント値レジスタＩＤ２０４の３つのフィールドから構成される。オペコード２０２は、本特殊命令と他の一般の命令を区別するための一意に決定されるコードであり、命令シミュレータ１００においてのみ意味のあるコードである。一般に、本命令が実機で実行された場合は、ノット・オペレーション（ＮＯＯＰ）になるようにコードが割り付けられる。イベント種類フィールド２０３は、本命令によって発生タイミングをコントロールしようとする対象となる非同期イベントの種類を表す一意の番号等が設定されるフィールドである。カウント値レジスタＩＤフィールド２０４は、本命令の実行から非同期イベントの発生タイミングまでの命令シミュレータ１００上での実行命令数を保持するカウント値レジスタの番号（汎用レジスタ番号など）が設定されるフィールドである。

【００１８】操作者は非同期イベントを発生させたいポイントより前に実行される命令列の中に特殊命令２０１を挿入する。その際、発生させたい非同期イベントの種類をイベント種類フィールド２０３に、また、本特殊命令から何命令実行後にイベントを発生させるかを保持したカウント値レジスタ（例えば、汎用レジスタを使用する）のＩＤ番号をカウント値レジスタＩＤフィールド２０４に指定する。特殊命令処理手段１０２は、被テスト命令列２００に埋め込まれた特殊命令２０１を実行して、カウント値レジスタＩＤフィールド２０４により該当するレジスタからカウント値を取得し、イベント種類フィールド２０３のイベント種類とともにタイミング制御処理手段４００に渡して、該タイミング制御処理手段４００を起動する。なお、カウント値レジスタは命令シミュレータ内に複数設け、ユーザのオペレーションによって自由に設定できるほか、カウント値レジスタにカウント値を設定する特殊命令を用意し、これを被テストプログラムに予め埋め込み、実行することによっても設定可能である。

【００１９】以上のように、被テスト命令列２００に、非同期イベントの発生タイミングを規定する情報を含んだ命令オペランドを持った特殊命令２０１を埋め込むことによって、任意のタイミングでのイベントの発生を指示することが可能となる。

7

【００２０】図３に、非同期イベントのイベントデータ３００の一例を示す。イベントデータ３００は、命令シミュレータ１００とは独立に実行されるイベント登録手段１０５によって生成・登録される。なお、イベントデータ３００はオペレータ自身がキーボード等から入力コマンドにより登録してもよい。

【００２１】イベントデータ３００は、イベントＩＤ３０１と、イベント種類３０２、イベント管理情報３０３、リソースデータ３０４の各フィールドより構成される。イベントＩＤ３０１は、各イベントに固有の番号であり、命令シミュレータ上での結果解析や表示に使用される。イベント種類３０２は、そのイベントの種類を示す値であり、特殊命令２０１の命令オペランドであるイベント種類フィールド２０３の値に対応する。イベント管理情報３０３は、本イベントを発生させたイベント登録処理手段１０５の情報等を保持し、命令シミュレータ上での結果解析や表示に使用される。リソースデータ３０４は、本非同期イベントの発生によって書き換えられる命令シミュレータ上のリソースとその書き換えられるデータの組合せの集合である。

【００２２】以上のように、イベントデータ３００は、発生すべき非同期イベントの具体的な情報を格納している。従って、後述するように、イベント発生処理手段５００は、これらの情報を使って命令シミュレータ１００上に指定された非同期イベントを発生させることが可能となる。

【００２３】図４に、タイミング制御処理手段４００の処理フローを示す。タイミング制御処理手段４００は、特殊命令処理手段１０２からの起動指示と、命令シミュレータ１００からの一命令シミュレーションが実行する毎に発生する終了信号を入力とする。また、イベントデータ３００を入力とし、特殊命令２０１の命令オペランドで指定されたタイミングで非同期イベントが発生するように、発生タイミングの調整を行う。

【００２４】タイミング制御処理手段４００は、命令シミュレータ１００において被テスト命令列２００中に埋め込まれた特殊命令２０１が実行されるタイミングで、特殊命令処理手段１０２によって起動（コール）される。

【００２５】タイミング制御処理手段４００では、まず、処理４０１において、特殊命令処理手段１０２から非同期イベントの種類とカウント値を受け取る。ここで、カウント値は、前記特殊命令から非同期イベントを発生させる時点までの実行命令数、即ち、現時点から非同期イベントを発生させるまでのタイミングを示している。該カウント値（実行命令数）をタイミング制御処理手段４００が保持するカウンタに初期セットする。次に、処理４０２において、命令シミュレータ１００の命令解釈実行処理手段１０１より入力される１命令のシミュレーションが終了したことを示す信号に同期して、前

8

記カウンタを１減算する。次に、処理４０３によって条件分岐を行い、カウンタの値が“０”でなければ、処理４０２に戻り、“０”であれば次の処理４０４に進む。カウント値が“０”の場合、既に命令シミュレータ１００は非同期イベントを発生させるべきタイミングに命令処理が進んでいることを意味するため、処理４０４において、非同期イベントを発生させるために命令シミュレータ１００の実行を停止させる。

【００２６】続いて、タイミング制御処理手段４００では、処理４０５において、イベントデータ３００を読み込む。次に、処理４０６において、処理４０１で得られた特殊命令２０１で指定された非同期イベント（発生させるべき非同期イベントの種類）が当該イベントデータ３００内に存在するか否かを調べる。その結果より、処理４０７において条件分岐が発生する。即ち、指定された種類の非同期イベントの情報がイベントデータ３００内にない場合は、処理４０６に戻り、再度イベントデータ３００内に対象となる種類の非同期イベントの情報の有無を調べる。一方、イベントデータ３００内にある場合は、次の処理４０８に進み、発生すべきイベントの種類を指定してイベント発生処理手段５００をコールする。

【００２７】以上のように、タイミング制御処理手段４００は、被テスト命令列２００に埋め込まれた特殊命令２０１の命令オペランドに従って、まず、処理４０２、４０３によって、命令シミュレータ１００での命令処理を指定されたタイミング（命令数）まで実行させる。次に、処理４０６、４０７によって、指定された非同期イベントの情報がイベントデータ３００に登録されるまで待ち合わせを行う。こうして特殊命令２０１によって指定されたタイミングに合わせて、非同期イベントを発生させることが可能となる。

【００２８】図５に、イベント発生処理手段５００の処理フローを示す。イベント発生処理手段５００は、タイミング制御処理手段４００から起動（コール）される。また、イベントデータ３００を入力とし、命令シミュレータ１００内のリソース管理処理手段１０３に働き掛け、指定された非同期イベントを命令シミュレータ１００上に発生させる。

【００２９】イベント発生処理手段５００では、まず、処理５０１において、イベントデータ３００より発生させるべき非同期イベントの情報を読み込む。ここで、該当する非同期イベントの情報は、タイミング制御処理手段４００によって既にイベントデータ３００内に存在することが保証されている。また、同じくタイミング制御処理手段４００により、命令シミュレータ１００が非同期イベントを発生させるタイミングで停止していることも保証されている。そこで、処理５０２において、イベントデータ３００のリソースデータ３０４を読み取り、命令シミュレータ１００上のリソース管理処理手段１０

3を使って、命令シミュレータ１００上の該当するリソースの値を書き換えることにより、命令シミュレータ１００上に非同期イベントを発生させる。最後に、処理５０３において、命令シミュレータ１００にシミュレーション開始信号を送り、命令シミュレーションを再開させる。

【００３０】以上のように、イベント発生処理手段５００は、イベントデータ３００から発生させるべき非同期イベントの情報を読み込み、これをリソース管理処理手段１０３に渡すことにより、命令シミュレータ１００上に非同期イベントを発生させることが可能となる。

【００３１】以上のように、図１の命令シミュレーションシステムでは、被テスト命令列２００に予め埋め込まれた特殊命令２０１の命令オペランドで指定された非同期イベントを、同じく命令オペランドで指定されたタイミングで命令シミュレータ１００上に発生させることが可能となる。具体的には、タイミング制御処理手段４００が、特殊命令処理手段１０２から特殊命令２０１のオペランドデータを受け取り、まず、命令シミュレータ１００を指定されたタイミングの命令まで実行させた後、外部から非同期に生成されてイベントデータ３００に格納される非同期イベントの情報を待ち合わせる。次に、非同期イベントの情報がそろった時点で、タイミング制御処理手段４００がイベント発生処理手段５００をコールし、初めてイベント発生処理手段５００がイベントを発生させる。これにより、非同期イベントの発生を、特殊命令の命令オペランドによって指定された任意のタイミングで発生させることが可能である。また、タイミング制御処理手段４００が指定されたタイミングで、命令シミュレータ１００の実行と非同期イベントの発生のタイミングの同期を取るため、再現性も保証される。さらに、対象非同期イベントや発生タイミングは、命令オペランドによって指定されるため、操作者は煩雑な操作を必要とせずに非同期イベントの発生をコントロールすることが可能となる。

【００３２】なお、図４に示すタイミング制御処理手段４００の処理フローでは、特殊命令処理手段１０２が特殊命令２０１を解釈実行し、タイミング制御処理手段４００は、該特殊命令処理手段１０２からその実行結果として発生すべき非同期イベントの種類およびカウント値を受け取るとしたが、命令解釈実行処理手段１０１自体あるいは特殊命令処理手段１０２が特殊命令２０１を解読したなら直ちにタイミング制御処理手段４００をコールし、該タイミング制御処理手段４００において、特殊命令２０１の命令オペランドを解釈・実行して、イベントの種類およびカウント値を取得することでもよい。

【００３３】図６に、本発明の第２の実施例である命令シミュレーションシステムの構成図を示す。本命令シミュレーションシステムでは、プロセス６１０とプロセス６２０で、それぞれ非同期に命令シミュレータ１００が

実行される。なお、これらのプロセスは、同一の計算機上で実行される必要はなく、ネットワーク等で接続された複数の計算機システムに分散されて実行されてもよい。

【００３４】図６において、プロセス６１０で実行される命令シミュレータ１は、図１で示した命令シミュレータ１００と同様の構成である。即ち、命令シミュレータ１は特殊命令処理手段１０２を備えた命令解釈実行処理手段１０１とリソース管理処理手段１０３を含み、それにタイミング制御処理手段４００及びイベント発生処理手段５００が付加される。これらの機能、動作は図１の実施例と同じであり、ここでの説明は省略する。

【００３５】本命令シミュレーションシステムでは、プロセス６１０と非同期に実行されるプロセス６２０上で別の命令シミュレータ２が実行されている。さらに、プロセス６２０にはイベント登録処理手段１０５が実装されている。命令シミュレータ２での命令実行中にプロセス６１０で実行されている命令シミュレータ１に伝達するイベントが発生した場合、イベント登録処理手段１０５が起動されて、そのイベントの情報を生成し、イベントデータ３００に登録する。一方、命令シミュレータ１で実行されている被テスト命令列１に埋め込まれた特殊命令によって指定されたタイミングで非同期イベントが発生するように、図１の実施例と同様にしてタイミング制御処理手段４００とイベント発生処理手段５００が動作し、命令シミュレータ１上に指定された任意のタイミングで非同期イベントが発生する。なお、イベントデータ３００は、複数のプロセス間で共有される記憶資源であり、ファイルとしての他、共有メモリやパケットとして実現されうる。

【００３６】以上のように、図６の命令シミュレーションシステムでは、非同期に動作する２つ以上の命令シミュレータ間でデータの入出力がある場合に、データを受け取る側の命令シミュレータの命令実行のタイミングと非同期に発生する他の命令シミュレータが生成する非同期イベントの発生タイミングを調整することにより、シミュレータ間の非同期動作の再現性を保証することができる。

【００３７】なお、図６においては、命令シミュレータ２から命令シミュレータ１への非同期イベントの伝播を示しているが、同様の機構を双方のシミュレータに持たせることにより、逆方向の非同期イベントの伝播も実現することが可能である。

【００３８】

【発明の効果】本発明の命令シミュレーション方法およびシステムによれば、命令シミュレーシタの外界等（コマンド入力、別の命令シミュレータの処理等）に起因する非同期に発生する非同期イベントの発生を、被テスト命令列に特殊命令を埋め込むという簡単な操作により、任意のタイミングで一意的に発生させることが可能とな

11

る。また、同一の特殊命令下においては、常に同一のタイミングで非同期イベントが発生し、再現性が保証される。これらにより、非同期動作を伴うソフトウェアに対して、製造前に容易にテスト可能となる。さらに、非同期イベントの発生タイミングを細かく変化させてシミュレーションするような詳細なテストケースを完全に実施することが可能となり、ソフトウェアの信頼性の確保に大きく寄与することが可能となる。

【図面の簡単な説明】

【図１】本発明の命令シミュレーション方法を実施するシステムの一例を示す構成図である。

【図２】本発明にかかる非同期イベントの発生タイミング制御用特殊命令の一例を示す図である。

【図３】本発明にかかるイベントデータの一例を示す図である。

【図４】本発明にかかるタイミング制御処理手段の処理＊

12

＊フローの一例である。

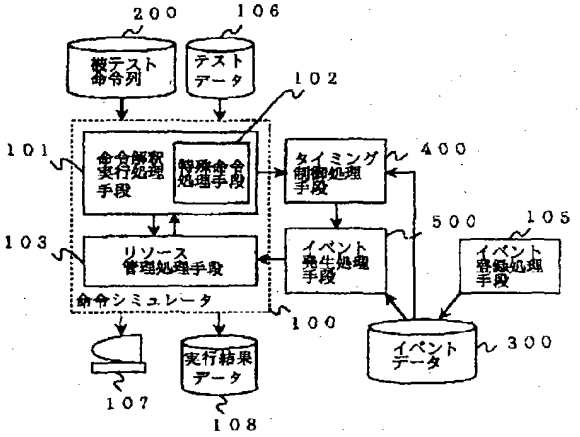
【図５】本発明にかかるイベント発生処理手段の処理フローの一例である。

【図６】本発明の命令シミュレーション方法を実施する第２のシステムの一例を示す構成図である。

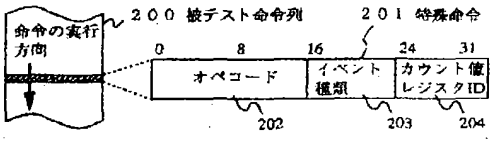
【符号の説明】

１００　命令シミュレータ
１０１　命令解釈実行処理手段
１０２　特殊命令処理手段
１０３　リソース管理処理手段
１０５　イベント登録処理手段
２００　被テスト命令列
２０１　特殊命令
３００　イベントデータ
４００　タイミング制御処理手段
５００　イベント発生処理手段

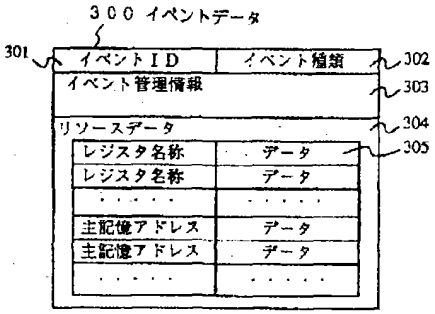
【図１】



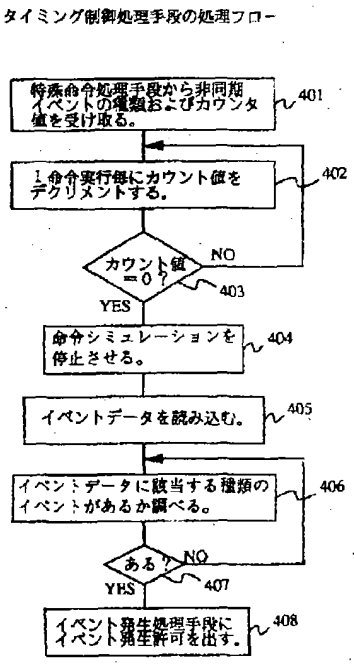
【図２】



【図３】

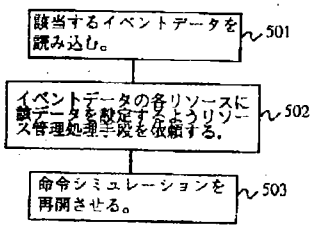


【図４】



【図5】

イベント発生処理手段の処理フロー



【図6】